ENTITY LINKING METHODS: TAGME VS FEL

Luca Lovagnini, Natural Language seminar.
University of Pisa.

INTRODUCTION

- Entity linking deals with identifying entities from a knowledge base in a given piece of text.
- TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities) *Paolo Ferragina (University of Pisa), Ugo Scaiella (University of Pisa)*.
- Fast and Space-Efficient Entity Linking in Queries, Edgar Meij (Yahoo Labs) *Giuseppe Ottaviano (ISTI-CNR), Roi Blanco (Yahoo labs) (FEL)*.
- We can identify three steps:
  1. Identifying candidate mentions, i.e., which part(s) of the text to link
  2. Identifying candidate entries for each mention
  3. Disambiguating the candidate entities based on some notion of context and coherence

Rolling Stones didn't play at Woodstock

- Jan 2015
- Specifically designed for search engine queries
- Efficency is crucial (usually not so considered in literature)
- Forward-backward scanning procedure implemented with dynamic programming in $O(k^2)$
- Candidates (*aliases*) are compressed thorugh hashing and bit encoding
- Probabilistic Model
- Fast Entity Linker (FEL) compute a probabilistic score for each segment-entity pair
- Wikipedia's anchor text used in FEL
- Yahoo's Webscope search query log
- (Almost) parameterless fashion (unlike TAGME)

FAST ENTITY LINKER

- *aliases:* textual representation of an entity (for example «rock» could be an alias both for stone or music genre)
- *SxE* event space where S are all the sequences and *E* the entities
- *s* is a sequence of terms *t*∈*s*
- **s** represents a segmentation (sequence of sequences of terms) where s ∈ **s** is drawn from the set S
- **e** represents a set of entities e ∈ e, where each e is drawn from the set E
- a_s indicates if *s* is an alias
- $a_{s,e}$ indicates if s is an alias pointing (linking/clicked) to e
- c indicates which collection acts as a source of information query log or Wikipedia (c_q or c_w)
- *n(s,c)* is the count of *s* in *c*
- *n(e,c)* is the count of *e* in *c*
- *q* input query
- S_q all possible segmentations of its tokens t_1, t_2, t_3, t_4 …

ANNOTATION

FEL – LINKING SEGMENTS TO ENTITIES

- Fel returns the set of entities e, along with their scores, that maximizes (notice entities indipendence assumption)

$$\underset{\boldsymbol{e}\in E}{arg\max} \log P(e|q) = \underset{\boldsymbol{e}\in E,\boldsymbol{s}\in S_q}{arg\max} \sum_{e\in\boldsymbol{e}} \log P(e|s) \quad (1)$$

- A possible alternative to (1) would be to select the segmentation that optimizes the score of the top ranked entity:

$$\underset{\boldsymbol{e}\in E}{arg\max} \log P(e|q) = \underset{\boldsymbol{e}\in E,\boldsymbol{s}\in Sq}{arg\max} \max_{e\in\boldsymbol{e},s\in\boldsymbol{s}} P(e|s) \quad (2)$$

FEL - LINKING SEGMENTS TO ENTITIES

- Now we define the probabilty that a segment s is linked to an entity

s is in c $\frac{n(c,s)}{\sum_{c'} n(c',s)}$

$$P(e|s) = \sum_{c \in \{cq,cw\}} P(c|s)P(e|c,s)$$

Knowing that s is in c, then it points to e

$$= \sum_{c \in \{cq,cw\}} P(c|s) \sum_{a_s=\{0,1\}} P(a_s|c,s)P(e|as,c,s)$$

alias s in c points to e

s in c is an alias

$$= \sum_{c \in \{cq,cw\}} P(c|s)\ [\cancel{P(a_s = 0|c,s)P(e|as = 0,c,s)} + P(a_s = 1|c,s)P(e|as = 1,c,s))]$$

s is an alias in c

alias s in c points to entity e

CONSIDERING MAXIMUM LIKELIHOOD!

$$\frac{\sum_{s:as=1} n(s,c)}{n(s,c)} \qquad \frac{\sum_{s:as_e=1} n(s,c)}{\sum_{s:as=1} n(s,c)}$$

- The maximum likelihood probabilities reported in the previous slide can be smoothed (estimated) with an add-one, Dirichlet and Laplace smoothings.
- Some term of the input query might not be covered by any segment *s*, so it was defined a special entity *not_linked* where *P(not_linked | s)=l* where *l* is a parameter which value depends on how many links we want to obtain from the algorithm.

FEL - LINKING SEGMENTS TO ENTITIES

FEL - GENERAL PROBLEM

- Let $t = t_1, t_2, t_3, \ldots, t_k$ a sequene of terms and let $[t_i, \ldots, ti_{+j}], \forall i, j \geq 0$ any segment of the sequence
- Let $\gamma(s)$ be any scoring function that maps segment to real numbers
- Let Φ(a,b) an aggregation funciton (Φ(a,b)=a+b for (1) and Φ(a,b)=max(a,b))
- Then the maximum score of a segmentation is defined as follows:

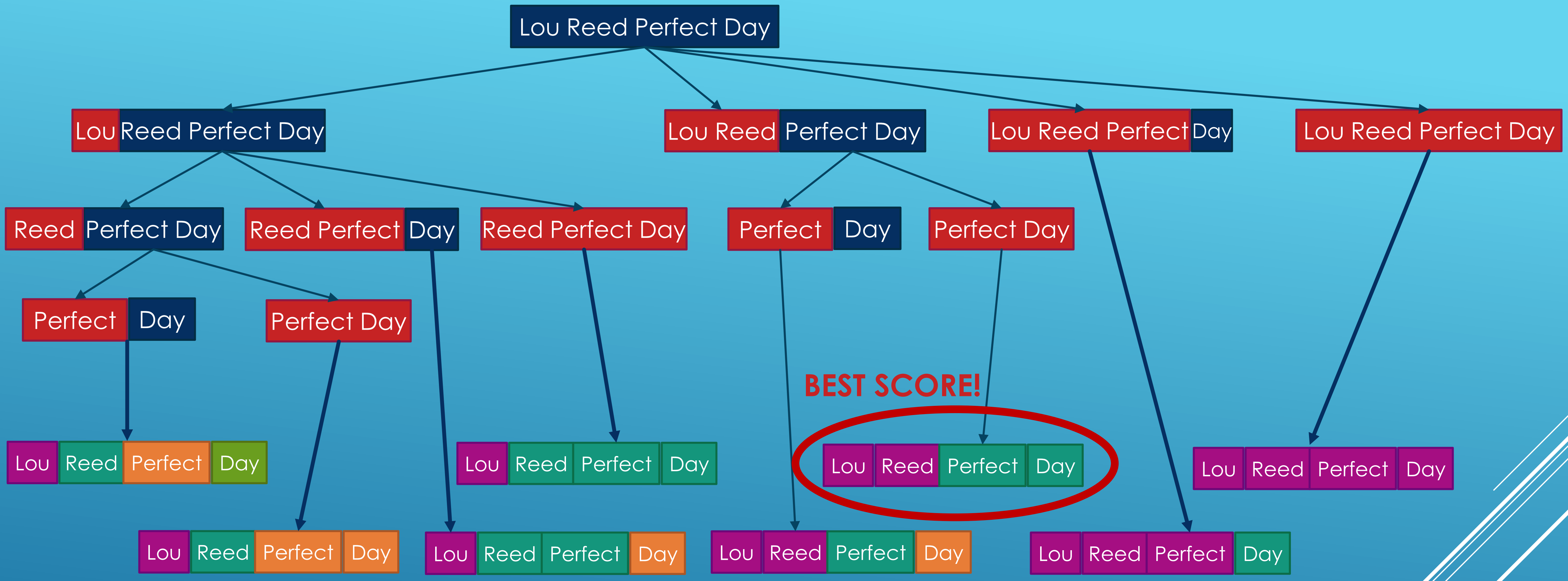
$$m(t_1, t_2, \ldots, {}_{tk}) \\ = \max(\Phi\big(\gamma(t_1), m(t_2, \ldots, tk)\big), \phi\big(\gamma([t_1, t_2]), m(t_3, \ldots, tk)\big), \ldots \phi\big(\gamma([t_1, \ldots tk$$

Algorithm 1 Entity-linking algorithm

Require: A user query q, a function HIGHESTSCORE$(\cdot)$, and an aggregation function $\phi(\cdot, \cdot)$.

```
p ← TOKENIZE(q)
l ← LENGTH(p)
maxscore[] ← new array[l + 1]
previous[] ← new array[l + 1]
for i = 0 to l do
    for j = 0 to i do
        score ← φ(maxscore[j], HIGHESTSCORE(p[j : i], q))
        if score > maxscore[i] then
            maxscore[i] ← score
            previous[i] ← j
        end if
    end for
end for
return maxscore[l]
```

FEL – GENERAL ALGORITHM



FEL – AN EXAMPLE

FEL – MODELING CONTEXT

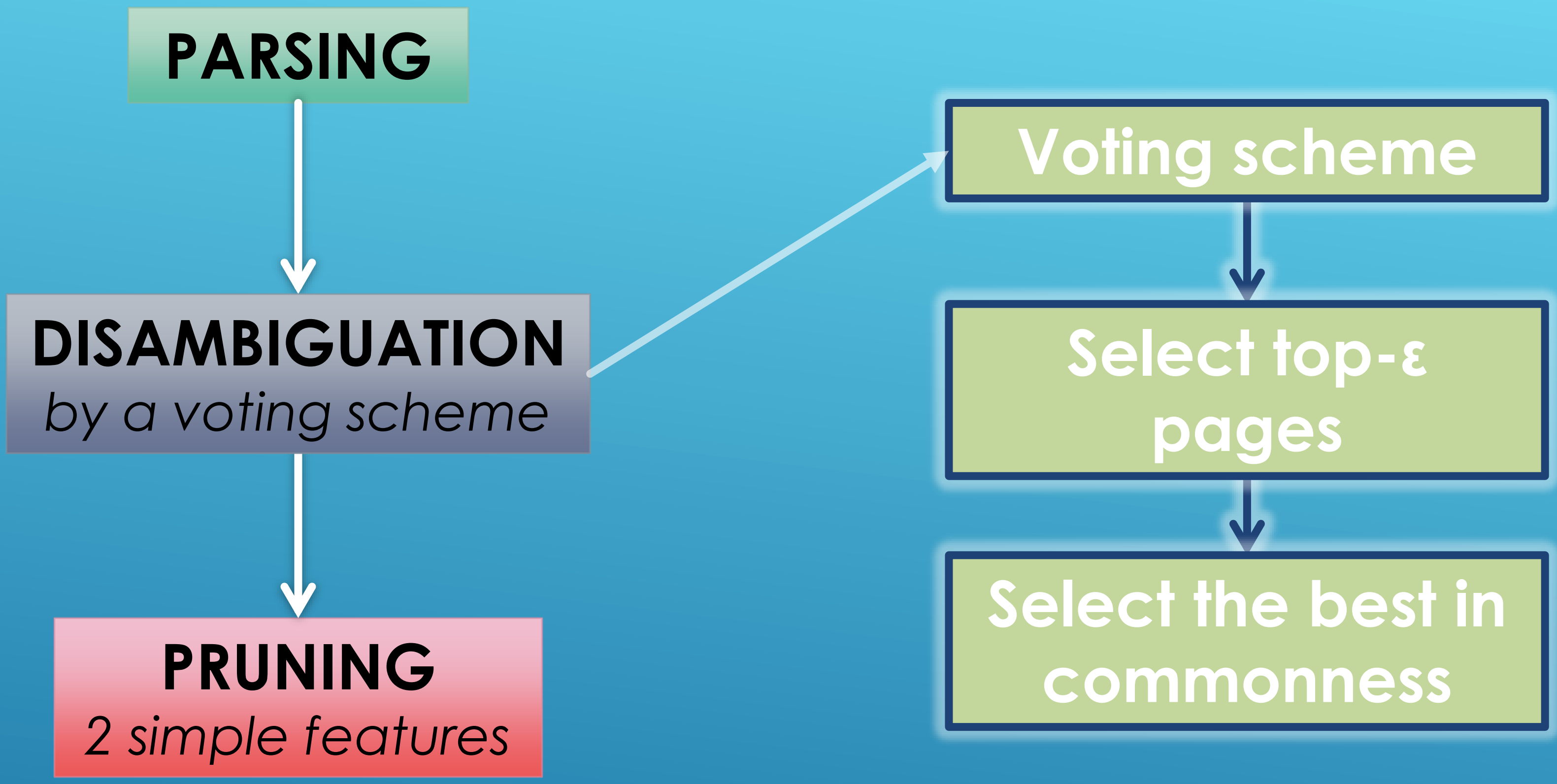
- Cannot distinguish «Brad Pitt seven» from «Brad Pitt olympics»
- The *contextual relevance model* estimates the probability that an entity e is relevant to the whole query (the context)
- $\gamma(\mathrm{s}) = \max_{e \in E} \log(P(e|s,q) = P(e|s) \prod_i \frac{P(ti|e)}{P(q)}$ *where* $P(t|e)$ is the probability that a term *t* is relevant to e
- Supposing the distributional semantics hypothesis (if t_1 is relevant to e and t_2 similar to t_1, then t_2 is relevant to e as well), a the Google's *word2vec* tool (3 bilion of words from Wikipedia) is used so words that are close in meaning are mapped to vectors close in cosine distance.
- So we define $P(t|e) = \sigma([\mathrm{v_t}\ 1]\ \mathrm{ve}])$ where $v_t, ve \epsilon \mathbb{R}^D$ are the vectors relative to the word *t* and the entity *e* (where D is usually 200) and $\sigma(x) = \frac{1}{1+e^{-x}}$
- Without going too much in details (for time constraints) if *(D=200)* then the complexity to score each entity *e* with a query is $O(kD)$ and the space needed would be $4(E(D+1)+WD)$ (vector compression implemented)
- Centroids can be used to obtain $O(D)$ (negligible improvment)

- 2010
- Google Research Awards winner
- Considered the «state-of-the-art» about entity linking
- Uses Wikipedia as knowledge base
- High precision/recall, annotates short fragments of text with pertinent hyperlinks to Wikipedia articles.
- The main feature is that it may annotate texts which are short and poorly composed, such as snippets of search-engine results, tweets, news, etc.

TAGME

- The spots are the sequence of terms in the input text which are to be annotated.
- Wikipedia anchor texts as spots and pages linked to them in Wikipedia as their possible senses (about 3 millions as dictionary).
- Ambiguity and polysemy solved between the potentially many available anchor-page mappings by finding the collective agreement among them via new scoring functions
- The time complexity of TAGME's annotation is linear in the number of processed anchors

TAGME



TAGME WORKFLOW

- Wikipedia page ***p***
- Text ***a*** of *p* (maybe an anchor)
- ***Pg(a)*** is the set of all Wikipedia pages linked by *a* (same anchor may occour many times pointing to different pages)
- ***freq(a)*** number of times that *a* occours in Wikipedia (as an anchor or not)
- ***link(a)*** number of times that *a* occours as an anchor in Wikipedia (so $link(a) \leq freq(a)$)
- ***lp(a)*** = *link(a)/freq(a)* to denote the *link-probability* that an occurrence of *a* has been set as an anchor (for example: lp(house) low < lp(Barack)).
- ***Pr(p|a)*** is the *commonness*, so the probability that an occurrence of an anchor *a point to* $p \in Pg(a)$ (for example: Pr(Apple Inc. page|«apple») > Pr(Apple fruit page| «apple»))
- $\boldsymbol{a \to p}$ represents the annotaion of an anchor *a* with some page $p \in Pg(a)$
- If $\boldsymbol{Pg(a) > 1}$ (so *a* has more senses), we call *disambiguation* the process of selecting one of the possible senses of a from *Pg(a)*.

ANNOTATION

- Short text as inputs
- Tokenization
- Anchor detection by quering the *Anchor Dictionary* for sequences up to 6 words
- Define «anchor boundaries»: if anchor *a1* is a substring of anchor *a2* we drop *a1* if *lp(a1)<lp(a2)* (since a1 is usually more ambigous)
- For example a1=«jaguar» a2=«jaguar car» (lp(a1)<lp(a2)): the anchor jaguar would slow down the process
- While a1=«act» a2=«the act» (band, small number of link occurences, lp(a1)>lp(a2): both words are kept because we're not able to make a principled pruning.

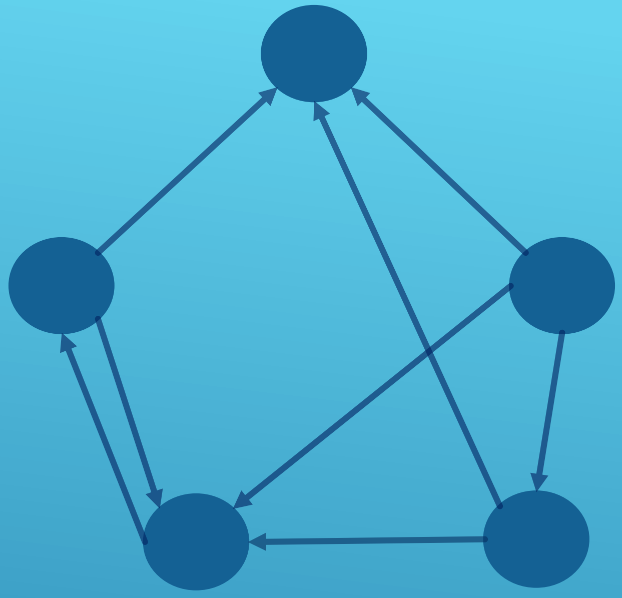
ANCHOR PARSING

- Let ***At*** be the set of all anchors occurring in a short input text ***T***
- For each anchor $\boldsymbol{a} \in At$ and for each possible sense of $\boldsymbol{p_a}$ of a, TAGME computes a score via a **«collective agreement»** between the sense p_a and the sense of all others anchors in *T*.
- The agreement score of $\boldsymbol{a} \rightarrow \boldsymbol{p_a}$ is evaluated by means of a ***voting scheme***, where each anchor *b* gives a vote (score) to p_a.

ANCHOR DISAMBIGUATION

ANCHOR DISAMBIGUATION - RELATENESS

- The vote that *b* gives to p_a is based on the ***average relatedness*** between each sense p_b of *b* and the sense p_a that we want to associate to a.
- $rel(p_a, pb) = \frac{\log(\max(|in(p_a), in(p_b)|)) - \log(|in(pa) \cap in(pb)|)}{\log(W) - \log(\min|in(p_a), in(p_b)|))}$
- Where *in(p)* is the number of Wikipedia pages pointing to page p and *W* is the total number of pages in Wikipedia
- Pages that contain both terms indicate relatedness, while pages with only one of the terms suggest the opposite.



HIGH CORRELATION!

LOW CORRELATION...

ANCHOR DISAMBIGUATION - SCORE

- Hence the vote given by an anchor *b* to the annotation $a \rightarrow pa$ is
- $vote_b(p_a) = \frac{\sum_{p_b \in Pg(b)} rel(p_b, pa) * \Pr(pb|b)}{|Pg(b)|}$
- *Pr(*p_b*| b)* is used because not all senses of b have the same statistical significance
- Finally, the total score for the annotation $a \rightarrow pa$ results
- $rel_a(p_a) = \sum_{b \in At \backslash a} vote(b, pa)$
- This score is combined with the commonness of the sense p_a
- All senses with a commonness smaller than a given threshold are discarded

- Now that we have computed the score for each possible sense of $p_a \in Pg(a)$, we have to chose the best one.
- First, we determine the sense p_{best} that achieves the highest relatedness $rel_a(p_{best})$ with the anchor a
- Then identifes the set of other senses in $Pg(a)$ that yield about the same value of $rel_a(p_{best})$, according to some fixed threshold ϵ
- Finally TAGME annotates a with the sense p_a that obtains the highest commonness $Pr(p_a | a)$ among these top- ϵ senses.

ANCHOR DISAMBIGUATION – CHOSING SENSE

- After the disambiguation phase, we have to *prune* the unmeaningful annotations from the set of candidate annotations.
- The goal of the pruning phase is to keep all anchors whose link probability (*lp*) is high or whose assigned sense (page) is ***coherent*** with the senses (pages) assigned to the other anchors.

ANCHOR PRUNING

- The coherence is based on the average relatedness between the candidate sense *pa* and the candidate senses p_b assigned to all other anchors *b*.
- We define as *S* the set of distinct senses, then the coherence
- $coherence(a \rightarrow pa) = \frac{1}{|S-1|} \sum_{p_b \in S \setminus \mathrm{p}_a} rel(pb, pa)$

ANCHOR PRUNING - COHERENCE

- For each candidate a ***pruning score*** is computed
- $\rho(a \rightarrow pa) = \frac{\mathrm{lp(a)+coherence}(a \rightarrow pa)}{2}$
- If $\rho\ (a \rightarrow pa) < \rho_{na}$(where $\rho_{na}$is a given threshold) then that annotation for *a* is discarded

ANCHOR PRUNING - THRESHOLD

Diego Maradona won against Mexico.

Which one?

- The text Diego isn’t necessary an anchor
- Instead Diego Maradona is generally an anchor
- lp(Diego)<lp(Diego Maradona) so the first one is discarded

TAGME EXAMPLE – ANCHOR PARSING

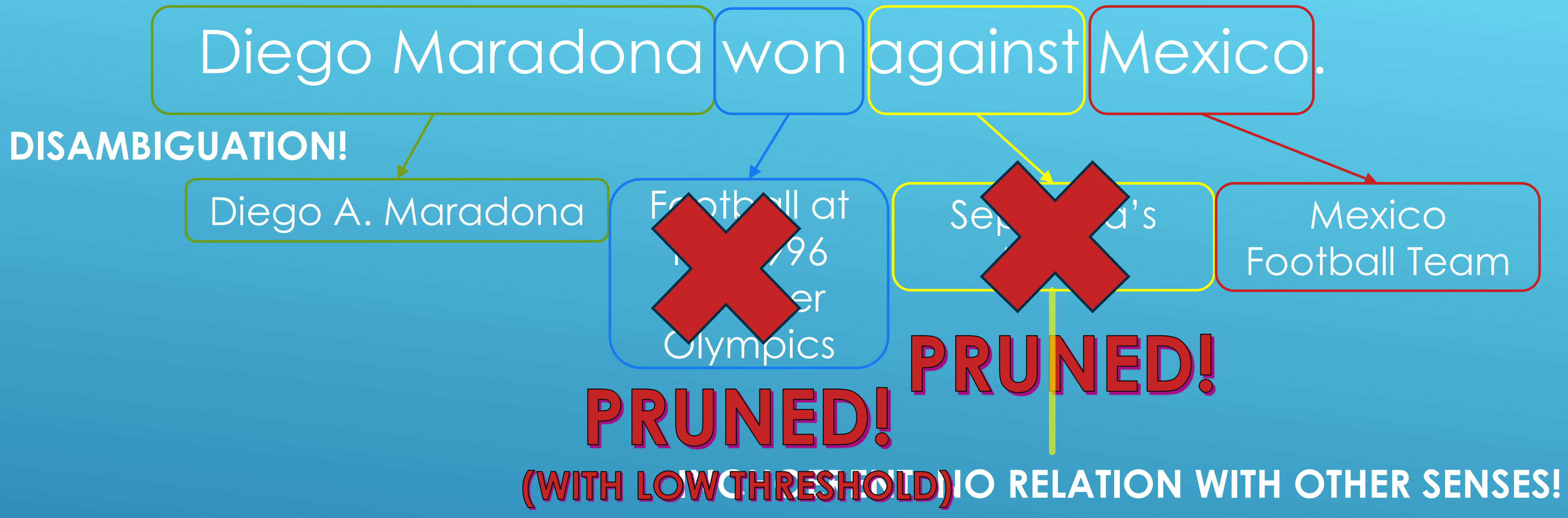
TAGME – DISAMBIGUATION

Diego Maradona won against Mexico.

Let's suppose that the disambiguation threshold ϵ is 0.6

Sense	Score	Pr(p | a): Commonness
~~Maradona Stadium~~	1.5	0.05
Diego A. Maradona	1.4	0.75
~~Maradona Film~~	1.3	0.05
~~Diego Maradona Jr.~~	0.5	0.15

TAGME - DISAMBIGUATION



TAGME EXAMPLE - PRUNING

TAGME

- $O(d_{in}(ns)^2)$ (d_{in}=avg in-degree Wikipedia's page, n number of anchors, s avg senses per anchor)
- Not specifically designed for efficency (totally in java!)
- Short text as inputs (more general)
- Only Wikipedia
- 3 phases could be bad: considered only certain entities
- Link probability for parsing
- Voting scheme for disambiguation
- Coherence for pruning
- Much more simple

FEL

- $O(k^2)$
- Efficency is crucial: optimization techniques implemented (early stopping and compression)
- Queries as input
- Wikipedia and Yahoo
- Probabilistic Model
- Each segment is indipendent
- Efficient through dynamic programming
- (Almost) parameterless
- Basic version without context
- Much more complicated

TAGME VS FEL

TAGME VS FEL

- From FEL paper
- Performance on the Yahoo's Webscope dataset
- 4 early preicision metrics: Precision at Rank 1, Mean Reciprocal Rank, Mean Avarage Preicison and R-Precision (bigger is better)
- Unfortunately, FEL efficency wasn't compared with TAGME: the AVG Time is taken from the two papers

	TAGME	FEL	FEL+LR	FEL+CENTROIDS
P@1	0.6682	0.7669	0.8352	0.8035
MRR	0.7043	0.8092	0.8684	0.8366
MAP	0.5458	0.6728	0.6912	0.6728
R-PREC	0.5462	0.6575	0.6883	0.6765
AVG ms	2 x anchor	0.14	0.4	0.27

- We have seen two different entity linking methods
- TAGME is older and introduced new concepts and imrpovments
- FEL is one of the newest methods, and was specifically designed to overall all the state-of-art methods and for web search engine queries
- Typical IR-based approach to indexing, clustering, classification and retrieval -> bag-of-words paradigm
- Natural Language Processing, machine learning, big data and concepts like context, relevance, relatedness are the new front-line to create a relation between user input text and entities.

CONCLUSION

QUESTIONS?